# Lab-7

## Linked-Stack

```
Struct Stack
{
  int data;
  Struct Stack *next;
};
Struct Stack *top = NULL;

void push() {
Struct stack *ptr;
ptr = (Struct Stack *) malloc (Size of (Struct Stack));
ptr->data = val; // val is from user
ptr->next = NULL;
if (top == NULL)
{
  top = ptr;
}
else
{
  ptr->next = top;
  top = ptr;
}
}

void display() {
  Struct stack *ptr;
  ptr = top;
  if (top == NULL)
    Printf("stack is empty");
```

```
else
{
    while(ptr != NULL)
    {
        Printf("%d \t", ptr->data);
        ptr = ptr->next;
    }
}
}

void pop(){
    Struct stack *ptr;
    ptr = top;
    if(top == NULL)
        Printf("Stack underflow");
    else{
        top = top->next;
        Printf("The value being deleted is: %d",
        free(ptr);                    ptr->data);
    }
}

int peep()
{
    return top->data;
}
```

## linked queue

```
struct node {
    int data;
    struct node *next;
};
node *root = NULL
```

~~struct queue {~~
~~struct node *front = NULL;~~
~~struct node *rear = NULL;~~
~~};~~
~~struct queue *q;~~

```
void insert() {
    struct node *ptr;
    ptr = (struct node *) malloc (sizeof (struct node));
    ptr -> data = val;
    ~~if (q -> front == NULL) {~~
    ptr -> next = NULL;
    if (root == NULL)
    {
      root = ptr;
    }
    else
    {
    Struct node *p = root;
      while (p -> next != Null)
      {
        p = p -> next;
      }
} } p -> next = ptr;
```

```
void dequeue()
{
struct node *temp;
  if(root==NULL)
  {
   printf("Queue is empty");
  }
 else
 {
   temp=root;
   root = temp->next;
   temp->next=Null;
   free(temp);
 }
}

void display(){ struct node *temp=ro
  while(temp!=NULL)
  {
   printf("%d \t",temp->data);
    temp = temp->next;
  }
}
```

## Reverse

```
void reverse(){
  Struct Node *prev=NULL;
  Struct Node *current=head;
  Struct Node *next=NULL;
```

```
while (current != NULL){
    next = current->next;
    current->next = prev;
    prev = currurrent
    current = next;
}
head = prev;
}
```

## Sorting

```
void bubbleSort(){
 int swapped, i;
 struct node *ptr1;
 struct node *lptr=null;
 if(head == NULL)
   return
 do {
    swapped=0
    ptr1=head
    while(ptr1->next != lptr)
    {
      if(ptr1->data > ptr1->next->data)
      {
        swap(ptr1, ptr1->next);
        swapped = 1;
      }
    ptr1 = ptr1->next
 }
iptr = ptr1; } while(swapped); }
```

## Condenation

```
void Concatenate (struct node *a, Struct node *b)
{
 if(a != NULL && b != NULL)
  {
    if(a->next == NULL)
        a->next = b;
    else
        Concatenate (a->next, b)
    }
    else
    {
      Printf("either a or b is NULL");
    }
Struct node *Concat(struct node * start1, struct
                                    node *start2)
{
  Struct node *ptr;
  if (start 1 == NULL {
      Start1 = Start 2;
       return start 1; }
  if (start 2 = NULL)
       return Start 1;
  ptr = Start 1;
  while (ptr -> link != NULL)
       ptr = ptr -> link;
  ptr -> link = start 2;
  return start 1;
}
```